Umask

Carefully restrict permissions for a file upon creation.

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-23

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 3984 bytes

Attack Category	• File Manipulation
Vulnerability Category	• Access Control • Unconditional
Software Context	• Security • File Creation • Temporary File Management
Location	• sys/stat.h
Description	umask() (the function) changes the umask (file creation permissions) of the current process. Permissions determine which users and groups can access a file. When a file is created, it must be assigned permissions to control who has access to it. The umask determines these initial permission. The way this assignment works is to take the set of permissions that corresponds to full access and to subtract those permissions set by the umask. The umask is important because if it is too loose, users may have access to files created by the process who should not have such access. If the mask is too tight, users in the process's group or users outside of that group may not have the access that was intended by the system designer.
APIs	**Function Name** / **Comments** umask()
Method of Attack	Processes running with a umask that is not restrictive enough will create files that are vulnerable to unintended reading or writing. An attacker could read those files using any pager or hexeditor and could edit them using an appropriate editor. The consequence could be disclosure of sensitive information, the surreptitious alteration of stored data, or the deletion of this data.
Exception Criteria	The only exception to the improper setting of a umask would be if the filesystem was inaccessible to

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

	any untrusted processes or users. This is rarely a safe assumption.
Solutions	**Solution Applicability**: Always. **Solution Description**: Always set the umask and always set it to the most restrictive value possible. **Solution Efficacy**: This solution should mitigate any risk of creating a file with overly-broad permissions.
Signature Details	mode_t umask(mode_t new_umask);
Examples of Incorrect Code	`//Open a file:` `int fp = fopen("output.dat", "w +");`

Examples of Corrected Code	`umask(S_IRWXG | SIRWXO); //Set the umask such that any files created won't allow the group or the world to read, write, or execute.`
	`int fp = fopen("output.dat", "w +"); //Open the file.`

Source References	• http://publib.boulder.ibm.com/infocenter/tpfhelp/v1r3m0/topic/com.ibm.tpf.doc_put.19/gtpc2/gtpc2m9o.htm#HDRRTUMA[2] • http://web.cse.msu.edu/cgi-bin/man2html?umask?1?/usr/man • Rough Auditing Tool for Security (RATS)[4]
Recommended Resource	
Discriminant Set	**Operating System**: • UNIX (All) **Languages**: • C • C++

Cigital, Inc. Copyright

1. mailto:copyright@cigital.com

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.